一种深度 Q 网络的改进算法 *

夏宗涛，秦 进

(贵州大学 计算机科学与技术学院，贵阳 550025)

摘 要：深度 Q 网络存在严重的过估计问题，导致智能体寻找最优策略的能力下降。为了缓解深度 Q 网络中存在的过估计问题，提出一个更正函数用于对深度 Q 网络中的评价函数进行改进，当选择的动作为最优动作时更正函数为 1，不对当前状态—动作值进行修改，当选择的动作不是最优动作时更正函数小于 1，缩小当前状态—动作值，从而使得最优状态—动作值与非最优状态—动作值的差异增大，减少过估计问题的影响。实验证明改进的算法在 Playing Atari 2600 视频游戏以及 OpenAI Gym 中取得了更好的性能。说明改进的算法比深度 Q 网络寻得了更优的策略。

关键词：深度 Q 网络；过估计问题；更正函数；状态-动作值

中图分类号：TP301.6 doi: 10.3969/j.issn.1001-3695.2018.07.0417

Improved algorithm for deep Q net

Xia Zongtao, Qin Jin

(*College of Computer Science & Technology Guizhou University*, *Guiyang* 550025, *China*)

Abstract: There is a serious overestimation problem in deep Q net, which leads to reduce the ability of the agent to find the optimal strategy. In order to relieve the overestimation in deep Q net, this paper proposed a correction function to improve the evaluation function of Deep Q Net. when the selected action is the optimal action, the correction function is 1, and the current state-action value is not modified. When the selected action is not the optimal action, the correction function is less than 1, and the current state-action value is reduced. Thus the difference between non-optimal state-action values increases, reducing the impact of overestimation. Experiments show that the improved algorithm achieves better performance in Playing Atari 2600 and OpenAI Gym. indicating that the improved algorithm could find a better strategy than deep Q net.

Key words: Deep Q Net; overestimation; correction function ; state-action value

0 引言

强化学习的基本思想是通过智能体不断的采取动作与环境进行交互，从而提高智能体对未知环境的适应能力，最大化从环境中所获得的奖赏，最终获得最优策略[1]。强化学习在人工智能领域里面有着广泛的应用，例如：工业制造[2]、路径规划[3]、游戏博弈[4]。传统的强化学习算法如 Q-learning[5]、Sarsa[6]都是利用 Q 表存储 Q 值，因此只能处理状态空间或动作空间较小的问题，在状态空间或动作空间较大的问题上，因为无法构建足够大的 Q 表，所以没有办法完整的表示状态空间以及动作空间。为了让强化学习算法能够处理状态空间以及动作空间都较大的问题，可以采用非线性函数评估模型实现对 Q 表的建模。DeepMind 在强化学习中引入了深度学习的思想[7]，提出了深度 Q 网络，将深度学习的感知能力和强化学习的决策能力相结合，解决了 Q 表无法覆盖完整的状态-动作空间的问题。深度 Q 网络利用卷积神经网络实现了对 Q 表的建模，从而完整的表示所有的状态-动作空间，在 Playing Atari 2600 视频游戏中取得了巨大的成功，在 49 个视频游戏中取得了超过人类专业玩家的成绩，但是将强化学习和类似神经网络的函数逼近结合会带来过估计问题[8]，造成最后的模型所选择的策略不是最优策略。Double DQN[9]通过将动作的选择和评价进行解耦合，从而缓和了过估计问题，但是却带来了欠估计问题[10]；Speedy Q-learning[11]通过结合第 k 步以及第 $k-1$ 步的状态-动作值来更新 Q 表，获得了更快的收敛速度，以及部分解决了过估计问题，但是遗憾的是，利用神经网络对 Speedy Q-learning 进行建模后，相比较深度 Q 网络，并没有获得更好的性能；平均深度 Q 网络[12]，不仅仅利用当前网络去获取目标值，而是利用过去 n 步的网络一同获取目标值并取平均，最后取最大值作为新的目标值，结果表明，平均深度 Q 网络取得了更高的平均奖赏值，但是却存在训练开销大，训练时间久的问题；Bias-Corrected Q-learning[13]，通过构造一个更正项来抵消 Max 算子所带来的平均误差，实验证明，在 roulette 游戏中取得了比简单的 Q-learning

收稿日期：2018-07-25；**修回日期：**2018-09-06 **基金项目：**国家自然科学基金（61562009）

作者简介：夏宗涛（1993-），男，山东省临沂市人，硕士研究生，主要研究方向为机器学习、强化学习（xiazongtao11@163.com）；秦进（1978-），男，贵州省黔西人，博士，副教授，主要研究方向为智能计算.

以及 Speedy Q-learning 更高的奖赏值，但是 Bias-Corrected Q-learning 更适合处理动作空间较大的问题，不适合处理动作空间普遍较小的 Playing Atari 2600 视频游戏，因此并不适合用来对深度 Q 网络进行改造；优势学习[14]可以在一定程度上缓解过估计问题，通过缩小非最优值，从而进一步保证最优值和次优值的次序不会因为估计误差以及 Max 算子的存在而发生改变，保证强化学习算法所学习到的策略是最优策略，但是优势学习对于不同非最优值的缩小是不合理的。本文设计一种新的方法，使得和最优值差值较小的非最优值得到较大的缩小，和最优值差值较大的非最优值得到较小的缩小，利用神经网络进行建模，应用到深度 Q 网络中产生改进算法：corrected-DQN（corrected deep Q net），实验证明，corrected-DQN 学习到了更优的策略，取得了更高的回报。

1 深度强化学习

1.1 强化学习

强化学习是一种从状态映射到动作的学习，通过与环境的交互，采集到真实环境中的样本，并从所采集到的样本中学习，以试错的方式学习到最优策略。强化学习系统一般包含智能体 agent 以及环境两个部分。智能体不断的与环境进行交互，在环境中执行动作，到达一个新的状态，并得到环境的奖励，直到智能体最终到达一个环境的吸收状态，此时不再选择动作，同时状态也不再发生改变。反复进行这个过程，使智能体最终学到一个最优策略，强化学习的决策过程基于马尔可夫决策过程[15]，通常假设强化学习任务满足马尔可夫性，一个马尔可夫决策过程可以定义为一个四元组 (S,A,P,R)，其中 S 为智能体所能达到的所有状态空间的集合； A 表示智能体所能选择的所有动作空间的集合，即所有可能的动作 $a\in A$；P 为状态转移概率，即智能体在一个状态下采取一个动作 a 后达到下一个状态的概率，记为 $P(s'|s,a)$；其中 s' 为状态 s 的下一步状态； R 为奖赏函数，智能体在状态 s 下采取动作 a 所达到下一个状态 s' 时所获得的立即奖赏，记为 $r(s,a)$；一般来讲， P 和 R 在一个强化学习任务中常常是未知的，需要智能体不断在环境中进行试错，不断地探索。

强化学习任务通常使用值函数来表示在一个状态上执行一个策略将来将会取得的累积奖赏，值函数可以分为两类，状态值函数以及状态-动作值函数，状态值函数用 $V(s)$ 来表示，表示从状态 s 开始，智能体所获得的累积期望回报，状态值函数 $V(s)$ 定义为

$$V^{\pi}(s)=\mathrm{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^{t}r_{t}\,|\,s_{0}=s\right] \tag{1}$$

其中：γ 为折扣因子，$\gamma\in[0,1]$，γ^{t} 表示随着 t 的增加，离当前状态越远的状态对累计奖赏的影响越来越小。状态-动作值函数用 $Q^{\pi}(s,a)$ 来表示，在状态-动作值函数中不仅知道初始状态 s，还知道初始状态下所采取的动作 a，定义为

$$Q^{\pi}(s,a)=\mathrm{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^{t}r_{t}\,|\,s_{0}=s,a_{0}=a\right] \tag{2}$$

强化学习中常常使用 Bellman 方程来解决马尔可夫决策问题，考虑到 Bellman 方程的状态值函数为

$$V^{\pi}(s)=r(s,\pi(s))+\gamma\sum_{s\in S}P(s'|s,\pi(s))V^{\pi}(s') \tag{3}$$

π 为智能体所采取的策略，在此为确定性策略，可以视为一个从状态空间到动作空间的映射，记 $a=\pi(s)$，表示在状态 s 下，执行策略 π 所得到的动作为 a 。

理想的策略应该最大化累积奖赏，此时的策略称为最优策略，一个强化学习任务可能有多个最优策略，记为

$$\pi^{*}=\arg\max_{\pi}\sum_{s\in S}V^{\pi}(s) \tag{4}$$

基于最优策略 π^{*} 的状态值函数为最优状态值函数，记为

$$V^{*}(s)=V^{\pi^{*}}(s)=\max_{a\in A}\sum_{s'\in S}P(s'|s,a)\left[r(s,a)+\gamma V^{\pi^{*}}(s')\right] \tag{5}$$

基于最优策略 π^{*} 的状态-动作值函数为最优状态-动作值函数，记为

$$Q^{*}(s,a)=Q^{\pi^{*}}(s,a)=\sum_{s'\in S}P(s'|s,a)\left[r(s,a)+\gamma\max_{a'\in A}Q^{\pi^{*}}(s',a')\right] \tag{6}$$

在传统的强化学习算法中，例如 Q-learning、Sarsa。通过建立一张 Q 表的方式来存储 $Q^{*}(s,a)$。这在小规模离散空间的强化学习任务上是适用的，但是当状态空间以及动作空间过于巨大时，受到硬件条件的限制，无法建立一张如此巨大的 Q 表存储 $Q^{*}(s,a)$，所以需要使用函数逼近对 Q 表建模，最流行的做法就是利用深度学习的思想实现对 Q 表的建模。

1.2 深度强化学习

深度强化学习采用神经网络作为函数近似的模型，通过强化学习反复调整神经网络的参数，最终实现神经网络对 Q 表的建模。深度强化学习通过将强化学习的决策能力和深度学习的感知能力结合起来，利用深度学习将大规模原始输入数据进行简单但是非线性的变换，转换为更高层次的抽象表达，从而发现数据内在的规律，然后通过强化学习，反复更新神经网络，使得神经网络能够较好的实现对 Q 表的拟合，智能体通过神经网络能够获得一个较为理想的策略。DeepMind 团队在 2013 最先提出深度 Q 网络[7]，成功实现了深度学习中的 CNN[16]和强化学习中的 Q-learning 的结合，在 49 个 Playing Atari 2600 视频游戏之中，取得了比人类玩家更高的得分。之后又出现了诸多变种，例如深度双 Q 网络[9]，深度循环 Q 网络[17]。

2 改进的深度强化学习算法

2.1 过估计问题

将神经网络和强化学习技术相结合会带来过估计问题，神经网络等函数评估模型的输出值含有估计误差，因此函数评估模型的估计值并不能真实的反映真实值，它们之间总是存在误差；同时，大部分的强化学习算法大都使用 Max 算子来选择当

前状态下的最优动作，当估计值与真实值之间的误差为均匀分布时，通过 Max 算子所选择的动作便有可能不是智能体在当前状态下所选择的最优动作，模型总是会倾向选择被放大的状态-动作值所对应的动作。在同一个状态所对应的不同状态-动作值差异较小的情况下，模型可能会将当前状态的非最优动作选出，从而导致模型最终无法学习到一个最优的策略，造成智能体的性能下降。

假设函数评估模型在状态 s 下采取动作 a 所带来的估计值为 $Q^{estimation}(s,a)$，目标值为 $Q^{target}(s,a)$，函数评估模型所带来的评估误差由 Y_s^a 表示，假设 Y_s^a 均匀分布在 $[-\varepsilon,\varepsilon]$ 上，ε 为误差的上界，可以得到以下证明[8]：

$$Q^{estimation}(s,a)=Q^{target}(s,a)+Y_s^a \tag{7}$$

估计值与真实值在状态 s 下的误差 B_s 为

$$\begin{aligned} B_s &=\left(r_s^a+\gamma max_{a'}Q^{estimation}(s',a')\right)-\left(r_s^a+\gamma max_{a'}Q^{target}(s',a')\right)\\ &=\gamma\left(max_{a'}Q^{estimation}(s',a')-max_{a'}Q^{target}(s',a')\right)\\ &=\gamma\left(max_{a'}\left(Q^{target}(s',a')+Y_{s'}^{a'}\right)-max_{a'}Q^{target}(s',a')\right)\\ &=\gamma\left(max_{a'}Y_{s'}^{a'}\right) \end{aligned} \tag{8}$$

因为 $Y_{s'}^{a'}$ 为 $[-\varepsilon,\varepsilon]$ 上的均匀分布，所以 $E[Y_s^a]=0$，所以 $P(max_aY_s^a>0)>P(max_aY_s^a<0)$。

结论：$当E[Y_s^a]=0 时，\quad \forall a\overset{often}{\rightarrow}E[B_s]>0$

即
$$Q^{estimation}(s,a)\overset{often}{>}Q^{target}(s,a) \tag{9}$$

估计值会经常大于真实值，即产生了过估计问题。

传统的 Q-learning 算法仅仅是通过计算最优状态值函数 $V^*(s)$ 来评估并选择当前状态下的最优动作，假设在状态 s 下，智能体只有 a_1,a_2 两个动作可以选择，其中 $Q(s,a_1)>Q(s,a_2)$，所以在状态 s 下，a_1 为最优动作，a_2 为次优动作。由于函数评估模型本身带有均匀误差 ε，所以可能出现以下情形：

$$Q(s,a_1)-\varepsilon<Q(s,a_2)+\varepsilon \tag{10}$$

此时通过 Max 选择最优动作，智能体会将次优动作 a_2，而不是最优动作 a_1 选择出来，最终智能体所学习到的策略将不是最优策略。

2.2　差值增长

一种改进的思路是通过降低次优值的大小将最优值和次优值之间的差距拉大，在真实值与估计值之间的误差相对较小的情况下，即便评估模型存在评估存在误差也不会影响最优值以及非最优值的次序，优势学习便是采用类似的思想，优势学习的定义为

$$A(s,u)=V^*(s)-\alpha\left(V^*(s)-Q^*(s,u)\right) \tag{11}$$

其中：u 为状态 s 下所采取的动作，未必是最优动作,公式中的第二项即为最优值和非最优值之间的差值，也是非最优值被缩小的值，可以看出，优势学习对于不同非最优值的缩小是不合理的，它对离最优值越远的状态-动作值缩小的程度越大，反而对最有可能代替最优值被 Max 算子选出的次优值缩小最小，这在一定程度上会降低差值增长的效果。所以本文设计一种新的方法，引入一个更正函数，在将所有的非最优值降低的前提下，使得和最优值差值较小的非最优值得到较大的缩小，和最优值差值较大的非最优值得到较小的缩小。

定义一个更正函数：

$$B(s,u)=\begin{cases}1 & 当Q(s,u)=\max\limits_{a\in A}Q(s,a)\\ b^{Q(s,u)} & 否则\end{cases} \tag{12}$$

其中：$b\in(0,1)$，由指数函数的性质可知，当底数 $b<0$ 时，随着 $Q(s,u)$ 的逐渐增大，$b^{Q(s,u)}$ 的值逐渐缩小，即和最优值越接近的非最优值的更正函数越小，当 $Q(s,u)=\max\limits_{a\in A}Q(s,a)$ 时，更正函数为 1，不对最优状态-动作值函数做任何改动，修改最优状态-动作值函数得到新的评价函数为

$$A(s,u)=V^*(s)\times B(s,u) \tag{13}$$

当 $Q(s,u)=\max\limits_{a\in A}Q(s,a)$ 时，$B(s,u)$ 为 1，$A(s,u)=V^*(s)$，并没有改动最优状态值函数，但是当 $Q(s,u)\neq\max\limits_{a\in A}Q(s,a)$ 时，$V^*(s)$ 的值会被缩小，从而使得非最优值与最优值之间的差距逐渐放大，因为神经网络的评估误差而存在的过估计问题便会因此而减弱，使得改进的最优状态值函数最终能够找到一个较好的策略。

2.3　改进的深度 Q 网络

DeepMind 于 2013 年首先提出了深度 Q 网络，最早 neural fitted Q-learning[18](NFQ)尝试使用类似深度 Q 网络的网络结构去解决一些简单的控制问题。但是由于 NFQ 是利用弹性反向传播(RPROP)更新神经网络的参数，同时由于神经网络的不稳定性，导致 NFQ 无法处理较为复杂的控制问题。深度 Q 网络利用目标神经网络以及回放记忆单元极大的提高了神经网络的稳定性，同时使用随机梯度下降(SGD)更新在线值网络，使得深度 Q 网络可以处理较为复杂的控制问题，同时深度 Q 网络在训练的过程中不需要添加额外的数据信息，直接将原始数据输入一个卷积神经网络，从而使得智能体对数据的处理更接近人，然后使用强化学习中的 Q-learning，通过与环境进行交互，得到带有奖赏值的样本数据，从而更新在线值网络的参数，经过一定的时间步，将在线值网络的参数复制到目标值网络，利用目标值网络计算目标值，在线值网络以及目标值网络的更新如图 1 所示。

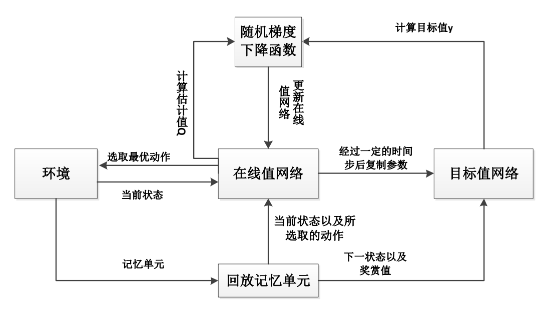


图 1　深度 Q 网络运行流程图

其中样本数据（记忆单元）的采集和使用通过回放记忆单元实现，一个记忆单元为一个四元组 (s,a,r,s')，其中 s 是当前的状态；a 是当前状态下所采取的动作；r 是智能体在状态 s 下采取动作 a 所获得的奖赏；s' 是智能体在状态 s 下采取动作 a 所到达的下一步状态。众多的四元组构成一个回放记忆单元，回放记忆单元中历史数据的利用采用随机抽样的方式，以此来打破数据之间的关联性。用抽样得到的数据训练卷积神经网络，采用在线值网络得到当前状态下的估计值，利用目标值网络表示当前状态下的目标值，计算当前状态下估计值和目标值之间的误差，利用随机梯度下降便可以更新在线值网络，模拟了 Q-learning 的迭代过程，也保证了神经网络的收敛性。

深度 Q 网络采用一个简化了的状态值函数求取目标值：

$$y=V^*(s;\theta')=r+\gamma\max_{a'}Q^*(s',a';\theta') \tag{14}$$

深度 Q 网络利用 θ 表示在线值网络，利用 θ' 表示目标值网络，为了使得本文提出的更正函数能够应用到深度强化学习算法中，必须对更正函数进行建模。

2.3.1 建模

在深度Q网络中，目标值网络的输出实现对目标值的建模，在线值网络实现对估计值的建模，本文所引入的更正函数是对目标值的更改，所以本文利用目标值网络对更正函数进行建模。将当前状态所对应的图像经过处理后输入目标值网络，在输出层，若 $Q(s,a;\theta')=\max_{a\in A}Q(s,a;\theta')$ 时，令更正函数为 1，若 $Q(s,a;\theta')\neq\max_{a\in A}Q(s,a;\theta')$ 时，在 $Q(s,a;\theta')>0$ 时，得到更正函数 $b^{Q(s,a;\theta')}<1$，即非最优值会被缩小。

$$A(s,u;\theta')=V^*(s;\theta')\times B(s,u;\theta') \tag{15}$$

u 为状态 s 下所采取的动作，来自于回放记忆单元中的历史数据，为在线值网络输出层的最大状态-动作值所对应的动作，即 $\max_{a\in A}Q^*(s,a;\theta)$，但是随着智能体与环境的不断交互，在线值网络的模型会被不断更新，历史模型所选择的结果未必和最新模型所选择的结果相同，这就为实现本文所提出的更正函数的建模带来了可能。

2.3.2 训练

利用本文提出的更正函数重新定义深度 Q 网络（DQN），得到改进的 DQN 算法（Corrected-DQN）：

初始化回放记忆单元 D，容量为 N，目标值网络的更新步长 C。

用随机权值 θ 初始化在线值网络，同时初始化目标值网络的权值 θ'，其中 $\theta'=\theta$。

初始化折扣因子 γ，更正函数的底数 b，episode=M。

重复（每一个 episode），直到 episode ≤ 0。

初始化状态 s，并且对于原始图像进行预处理得到 $\phi=\phi(s)$,episode=episode-1。

重复（对于每一个 episode 中的每一步）。

以概率 ε 随机选择一个动作 a。

否则以 $1-\varepsilon$ 的概率执行 $a=\arg\max_{a\in A}Q(s,a;\theta)$。

智能体在模拟环境中执行动作 a，观察得到的奖赏值 r 以及下一步的状态 s'，对于原始图像进行预处理得到 ϕ'。

将记忆单元 (ϕ,a,r,ϕ') 存入回放记忆单元 D 中，令 $u=a$。

从回放记忆单元 D 中随机抽取 minibatch 个记忆单元。

计算更正函数。

$$B(\phi,a;\theta')=\begin{cases}1 & \text{当}Q(\phi,a;\theta')=\max_{a\in A}Q(\phi,a;\theta')\\ b^{Q(\phi,a;\theta')} & \text{否则}\end{cases}$$

计算目标值。

$$y=A(\phi,a)=V^*(\phi)\times B(\phi,a)$$

其中目标值中的 $V^*(\phi)$ 为

$$V^*(\phi)=\begin{cases}r & \text{如果下一状态是结束态}\\ r+\gamma max_{a'}Q(\phi',a';\theta') & \text{其它}\end{cases}$$

计算损失函数 $(y-Q(\phi,a;\theta))^2$ 并通过梯度下降更新在线值网络的参数 θ。

每经过 C 步，令 $\theta'=\theta$。

结束循环。

结束循环。

3 实验结果与分析

3.1 实验设计

为了验证 Corrected-DQN 的性能，本文在 Arcade Learning Environment(ALE)以及 OpenAI Gym 两种实验环境下对 NIPS-DQN 和 Corrected-DQN 做对比实验。

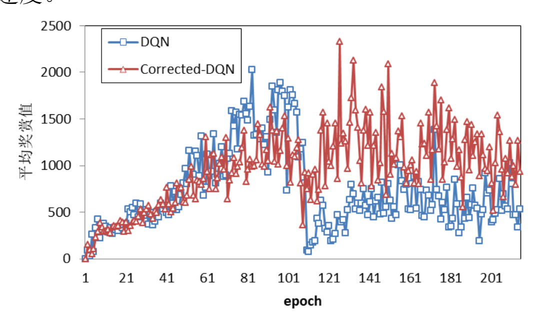
在实验环境 ALE 下，采用和 NIPS-DQN[7]完全相同的实验环境，同时为了提高计算速度使用 Tensorflow-GPU 1.2，GPU 型号为 Nvidia Quadro P6000，为了减少算法对算力的要求，本文直接使用文献[7]所述的 NIPS-DQN 算法作为 baseline。在 Playing Atari 2600 游戏中最为经典的控制问题：breakout、seaquest、phoenix、krull、amidar 上对算法 NIPS-DQN 以及 Corrected-DQN 进行了对比。本文采用原始论文中所给出的实验参数，其中回放记忆单元的大小初始化为 1000000，即可以存放 1000000 个样本数据；折扣因子初始化为 0.95，更正函数的底数 b 初始化为 0.95，以步长 0.05 减少到 0.5；贪心算法中进行探索的概率 ε 随着步长的增加由 1 逐渐减少到 0.1；算法的输入数据直接便是 Playing Atari 2600 视频游戏的原始图像，图像的大小为 210×210，同时带有 128 种不同的颜色，对图像进行预处理，生成大小为 84×84 的灰度图。设计双网络结构：在线值网络以及目标值网络。两个网络都是卷积神经网络，并且网络的结构完全一样，两者仅权值不同，目标值网络的权值周期性的拷贝于在线值网络，周期设置为 10000 步。在线值网络

由两个卷积层和两个全连接层构成，第一个卷积层有 16 个 8×8 大小的滤波器，步长为 4；第二个卷积层有 32 个 4×4 大小的滤波器，步长为 2，输入为 20×20×16 的向量空间，输出 9×9×32 大小的向量空间，激活函数为 ReLU；第一个全连接层有 256 个神经元；第二个全连接层神经元个数为动作空间的大小，输出状态 s 在各个可能采取的动作上的估计值。利用在线值网络得到当前状态的估计值，随机的从回放记忆单元抽取 32×4 个样本数据，以此打破样本数据之间的关联性。每次输入在线值网络的都是连续的 4 个状态联合的图像，每一个状态图像的大小为 84×84，状态所对应的图像经过预处理后输入在线值网络，输出层的输出对应的是可能的输出动作的状态-动作值，通过 Max 算子选择最大的状态—动作值所对应的动作为最优动作，智能体执行最优动作，从环境获得奖励，并且达到下一个状态，得到记忆单元所对应的四元组，将记忆单元按顺序插入回放记忆单元中。若回放记忆单元的空间已经被占满，则从第一个记忆单元开始覆盖之前的记忆单元，利用目标值网络求得目标值，将目标值与估计值相减，利用随机梯度下降更新在线值网络，每经过 10000 个时间步，将在线值网络的参数复制给目标值网络，通过目标值网络对更正函数进行建模，使得更正函数能够应用于深度 Q 网络。

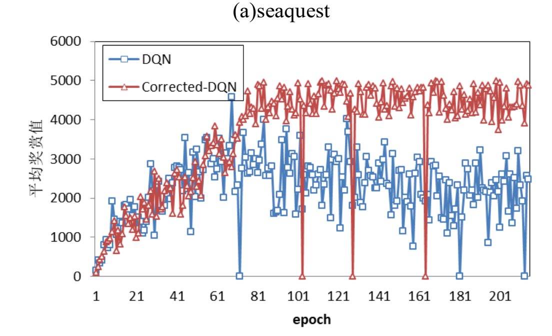
在实验环境 OpenAI Gym 下，由于 OpenAI Gym 与 ALE 存在诸多不同，所以本文对相应的实验参数进行调整。回放记忆单元的大小调整为为 10000，目标值网络的权值更新周期调整为 1000，每一个 epoch 调整为 10 000；同时由于 OpenAI Gym 中所选取的三个控制问题 Acrobot、CartPole 以及 MountainCar 所返回的状态不再是图像而是相关的特征（位置，速度等），问题规模大幅减少，所以不需要卷积层对原始数据进行降维，从而简化了网络结构，其余实验参数不改变；实验的过程中，无须进行图像处理这一步，直接将返回的状态输入网络即可，其余过程也不改动。

3.2 实验结果与分析

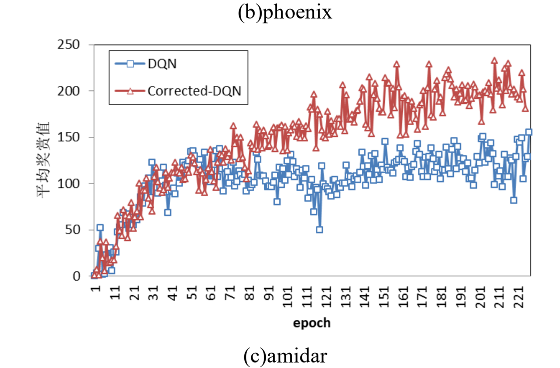
在 ALE 中，本文将 Corrected-DQN 和 NIPS-DQN 分别在控制问题：seaquest、phoenix、amidar、breakout、krull 进行实验，在平均奖赏值、算法的稳定性以及收敛速度三个方面对两种算法进行分析。平均奖赏值越高说明算法的性能越好，通过实验比较发现，Corrected-DQN 相比较 NIPS-DQN，在控制问题 seaquest、phoenix、amidar、breakout、krull 上取得了更高的平均奖赏值，说明改进的算法 Corrected-DQN 在控制问题 seaquest、phoenix、amidar、breakout、krull 上取得了更好的策略，智能体有着更好的性能；由于使用神经网络作为 Q 表的泛化模型，使得深度 Q 网络存在着不稳定的问题，为了缓解这个问题，深度 Q 网络采用了记忆回放机制以及双网络结构从而提高了深度 Q 网络的稳定性，但是从平均奖赏值较大的波动中可以看出，神经网络依然是不稳定的，通过实验发现，通过缓解过估计问题，智能体进行动作选择时有更大的概率选择出最优动作，Corrected-DQN 在 Playing Atari 2600 游戏中的控制问题 seaquest、phonix、krull 上取得了更加稳定的策略；在 krull 上有着更快的收敛速度。



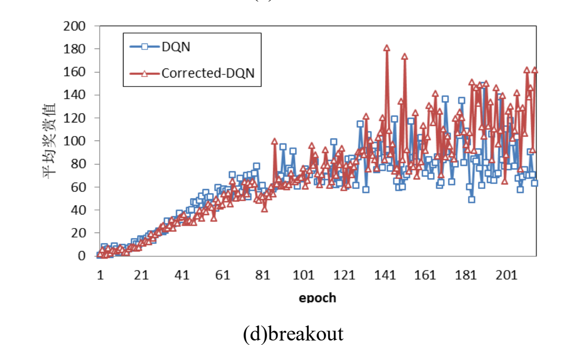
(a)seaquest



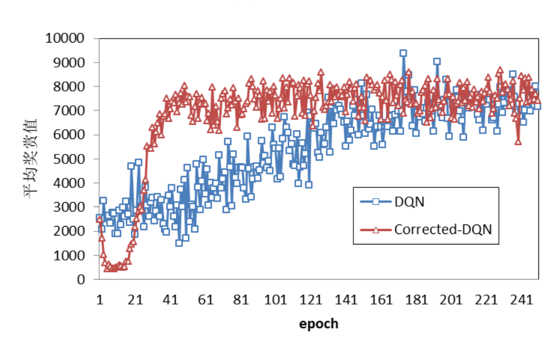
(b)phoenix



(c)amidar



(d)breakout



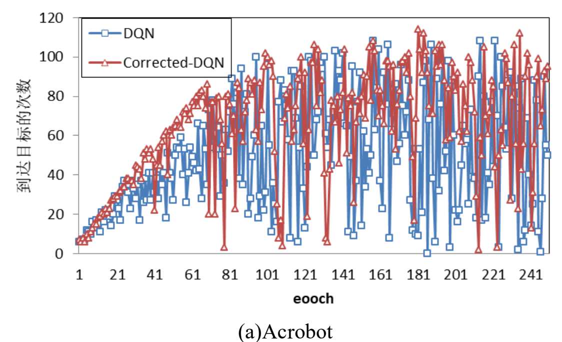
(e)krull

图 2　Corrected-DQN 以及 NIPS-DQN 在 ALE 中五种控制问题上的性能比较

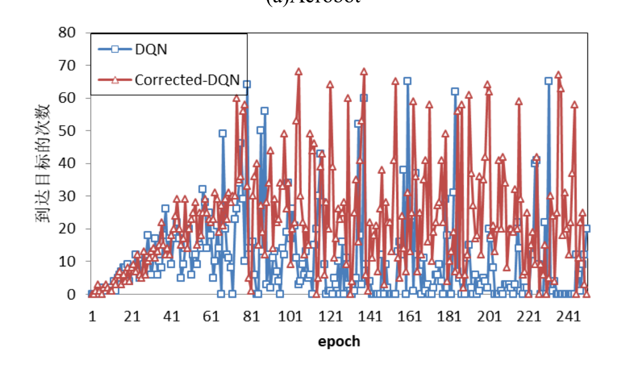
如图 2 中的(a)~(e)所示，Corrected-DQN 在控制问题

seaquest、phoenix、amidar、breakout、krull 上取得了比 NIPS-DQN 更高的平均奖赏值，说明相比较 NIPS-DQN，改进的算法 Corrected-DQN 的确获得了更优的策略；同时发现在控制游戏 krull 上 Corrected-DQN 取得了更快的收敛速度，仅仅通过 20 个 epoch（每一个 epoch 大概是 50000 步）便可以稳定在一个较高的水平；在算法的稳定性方面，通过在控制问题 seaquest、phoenix 以及 krull 上的实验对比可以发现，相比较 NIPS-DQN 的实验折线图，Corrected-DQN 的实验结果更趋平稳，说明在这几个控制问题上，神经网络的稳定性得到了进一步的提高。但同时本文发先无论是 Corrected-DQN 还是 NIPS-DQN 在控制问题 phoenix 上都会出现平均奖赏值为 0 的情况，这在一定程度上说明两种算法都还有很大的提高空间。

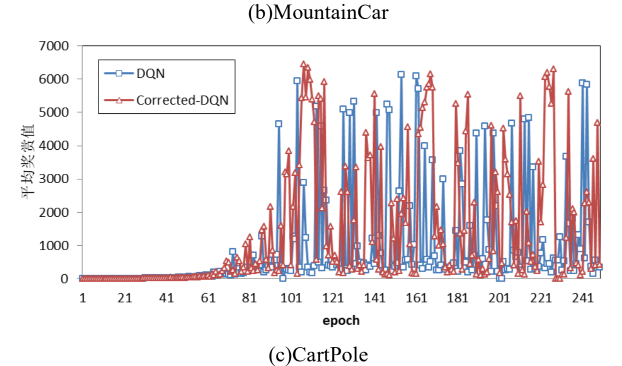
在 OpenAI Gym 中，本文将 Corrected-DQN 和 NIPS-DQN 分别在控制问题：Acrobot、CartPole 以及 MountainCar 进行实验，其中以平均奖赏值来评价 Corrected-DQN 和 NIPS-DQN 在控制问题 CartPole 上的性能，以一个 epoch（10000 步）智能体所到达目标的次数来评价 Corrected-DQN 和 NIPS-DQN 在控制问题 Acrobot、MountainCar 上的性能。



(a)Acrobot



(b)MountainCar



(c)CartPole

图 3　Corrected-DQN 以及 NIPS-DQN 在 OpenAI Gym 中三种控制问题上的性能比较

从图 3 中的(a)可以看到，在控制问题 Acrobot 中，改进的算法 Corrected-DQN 相比 NIPS-DQN 有着更好的稳定性，震荡的幅度降低，同时计算总共 250 次 epoch 的累计奖赏再求平均之后得到：Corrected-DQN 的平均奖赏值为 65，NIPS-DQN 的平均奖赏值为 50，说明 Corrected-DQN 也寻得了更好的策略。从图 3 中的(b)中可以看到，在控制问题 MountainCar 中，改进的算法 Corrected-DQN 相比 NIPS-DQN 达到了更多次目标，可以说明智能体最终寻得了更优的策略；但是在图 3 中的(c)中，并不能直观的看出 Corrected-DQN 和 NIPS-DQN 孰优孰劣，经过计算总共 250 次 epoch 的累计奖赏再求平均之后得到：Corrected-DQN 的平均奖赏值为 1401，NIPS-DQN 的平均奖赏值为 896。因此可以说明在控制问题 CartPole 上，Corrected-DQN 仍然学习到了更优的策略。但是同时本文发现，相比较实验环境 OpenAI Gym，Corrected-DQN 在实验环境 ALE 中性能提升的更加明显，初步分析认为，这是由于 OpenAI Gym 中所选取的控制问题相比较 ALE 中所选取的控制问题的输入规模大大降低所致，智能体无法获取更多的数据特征，从而导致算法的性能降低。

4　结束语

本文针对深度 Q 网络中存在的过估计问题，提出了一种基于差值增长的深度强化学习算法，通过设计一个新的更正函数，使得距离最优值越近的非最优值缩小的比例越大，反之，使得距离最优值越远的非最优值缩小的比例越小，从而使得最优值与非最优值的差值逐渐增大，最终降低非线性估计模型的评估误差对动作选择的影响，缓解了过估计问题，智能体学习到了更优的策略。

在 Playing Atari 2600 视频游戏以及 OpenAI Gym 中，改进的算法 Corrected-DQN 在平均奖赏值、算法的稳定性以及收敛性都比 NIPS-DQN 有一定程度的提高，因此可以得出结论，将本文提出的更正函数应用在深度 Q 网络中可以有效的缓解深度 Q 网络存在的过估计问题。

参考文献：

[1] 周志华. 机器学习 [M]. 北京: 清华大学出版社, 2016: 371-373. (Zhou Zhihua. Machine learning [M]. BeiJing: Tsinghua University Press, 2016: 371-373.)

[2] 高阳, 周如益, 王皓, 曹志新. 平均奖赏强化学习算法研究 [J]. 计算机学报, 2007, 29 (8): 2806-2810. (Gao Yang, Zhou Ruyi, Wang Hao, Cao Zhixin. Research on average reward reinforcement learning algorithm [J]. Chinese Journal of Computers, 2007, 29 (8): 2806-2810.)

[3] 杨文臣, 张轮. 多智能体强化学习在城市交通网络信号控制方法中的应用综述 [J]. 计算机应用研究, 2018, 35 (6): 1613-1618. (Yang Wenchen, Zhang Lun. Multi-agent reinforcement learning based traffic signal control for integrated urban network: survey of state of art [J]. Application Research of Computers, 2018, 35 (6): 1613-1618.)

[4] Tan M, Multi-agent reinforcement learning: independent vs. cooperative

agents [C]// Proc of the 10th International Conference on Machine Learning. San Francisco, CA: Morgan Kaufmann Publishing, 1993: 487-494.

[5] Peter D, *et al.* Q-learning [J]. Machine Learning, 1992, 8 (3): 279-292.

[6] Rummery G, Niranjan M. On-line Q-learning using connectionist systems [C]// CUED//F-INFENG//TR 166. England. 1994

[7] Mnih V, Kavukcuoglu K, Silver D, *et al.* Playing atari with deep reinforcement learning. arXiv preprint arXiv: 1312. 5602, 2013

[8] Thrun S, Schwartz A. Issues in using function approximation for reinforcement learning [C]// Proc of the 4th Connectionist Models. 1993

[9] Van H V, Guez A, Silver D. Deep reinforcement learning with double Q-learning [C]// Proc of the AAAI Conference on Artificial Intelligence. 2016: 2049-2100

[10] Hasselt H V, *et al.* Double Q-learning [C]// Advances in Neural Information Processing Systems. 2010: 2613-2621

[11] Azar M, Munos R, Ghavamzadeh M, Kappen H. Speedy Q-learning [C]// Advances in Neural Information Processing Systems, 2011

[12] Oron A, Nir B, Nahum S. Averaged-DQN: variance reduction and stabilization for deep reinforcement learning [EB//OL]. arXiv: 1611. 01929, 2016

[13] Lee D, Defourny B, Powell W. Bias-corrected Q-learning to control max-operator bias in Q-learning [C]// Proc of IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning. 2013: 93-99.

[14] Baird Ⅲ L C. reinforcement learning through gradient descent [D]. Carnegie Mellon University, USA. 1999

[15] 邱祎，董彦彦. 基于马尔可夫过程的线性规划方法探讨 [J]. 统计与决策, 2017, 10: 88-90. (Qiu Yi, Dong Yanyan. Linear programming method based on Markov process [J]. Statistics & Decision, 2017, 10: 88-90.)

[16] Lecun Y, Botton L, Bengio Y, Haffer P. Gradient-based learning applied to document recognition [J]. Proceedings of the IEEE, 1998, 86 (11): 2278-2324

[17] Hausknecht M, Stone P. Deep recurrent Q-learning for partially observable MDPs [EB//OL]. arXiv: 1507. 06527, 2015

[18] Martin R. Neural fitted q iteration-first experiences with a data efficient neural reinforcement learning method [C]// Machine Learning: ECML 2005, 317-328.